**WHAT IS CLAIMED IS:**

1. A method comprising the steps of:
loading device-independent driver code into kernel mode memory, wherein the device-independent driver code forms a first portion of a display driver;
receiving a device identifier associated with a particular device;
identifying a particular device-specific driver portion from a plurality of driver portions associated with the device identifier; and
loading the particular device-specific driver portion into kernel mode memory, wherein the device-specific driver portion forms a second portion of the display driver.

2. The method as in Claim 1, further including a step of requesting a device identifier, wherein the device identifier is to identify the particular device, after the step of loading device-independent driver code into kernel mode memory and before the step of receiving the device identifier.

3. The method as in Claim 1, wherein the device identifier includes an application-specific integrated circuit identifier.

4. The method as in Claim 3, wherein the hardware identifier includes a graphics chip identifier.

5. The method as in Claim 1, wherein the hardware-specific driver portion includes direct draw functions.

6. The method as in Claim 1, wherein the hardware-specific driver portion includes direct 3D functions.

7. The method as in Claim 1, wherein the step of loading the hardware-specific driver portion includes a step of calling a function to load a block of executable code in kernel mode memory.

8. The method as in Claim 7, wherein the function includes EngLoadImage function.

9. The method as in Claim 8, further including a step of identifying addresses of functions associated with the device-specific driver portion through a EngFindImageProcAddress function, after the step of loading the device-specific driver portion into memory.

10. The method as in Claim 1, wherein the device-independent driver code includes two-dimensional graphics functions.

11. The method as in Claim 1, wherein the step of identifying the device-specific driver portion includes locating a name associated with the device-specific driver portion in a table using the device identifier.

12. The method as in Claim 1, further including a step of comparing versions associated with functions of the device-specific driver portion to versions expected through an application program interface.

13. A method comprising the steps of:
providing a set of device-independent functions, wherein the device-independent functions are capable of supporting a plurality of different display devices;
providing a plurality of device-specific driver portions, wherein each device-specific driver portion of the plurality of device-specific driver portions include functions only capable of supporting a portion of the plurality of different display devices;
providing a first function to request for a device identifier, wherein the device identifier is capable of identifying a particular display device of the plurality of different display devices; and
providing a second function to load a particular device-specific driver portion into kernel mode memory, wherein the particular device-specific driver portion is associated with the particular display device of the plurality of different display devices.

14. The method as in Claim 13, wherein the device-independent functions include two-dimensional graphics processing functions.

15. The method as in Claim 13, wherein the second function includes a call to an EngLoadImage function.

16. The method as in Claim 13, further including a step of providing a third function to determine addresses associated with functions of the particular device-specific driver portion, after the step of providing the second function call.

17. The method as in Claim 16, wherein the third function includes a call to an EngFindImageProcAddress function.

18. The method as in Claim 13, wherein functions of the plurality of device-specific driver portions include direct 3D functions.

19. The method as in Claim 13, wherein functions of the plurality of device-specific driver portions include direct draw functions.

20. The method as in Claim 13, wherein the device identifier includes a graphics processor identifier.

21. The method as in Claim 13, wherein the device identifier includes an application specific integrated circuit identifier.

22. The method as in Claim 13, further including providing a table linking device identifiers to individual device-specific driver portions of the plurality of device-specific driver portions.

23. A system comprising:

a data processor having an input/output buffer;

memory having an input/output buffer coupled to the input/output buffer of the data processor, said memory having:

a kernel mode memory including:

a miniport driver to

initialize a display driver to be accessed as a portion of said kernel mode memory;

load device-independent driver code into said display driver in said kernel mode memory;

determine a device identifier associated with a display adapter;

identify device-specific driver code from a plurality of executable images, wherein the device-specific driver code is associated with said device identifier;

load a portion of device-specific driver code for access as a portion of said display driver;

said display driver, wherein said display driver includes:

said device-independent driver code;

said device-specific driver code;

said plurality of executable images;

display adapter having:

an input/output buffer coupled to the input/output buffer of the data processor; and

said device identifier.

24. The system as in Claim 23, wherein the device identifier includes an application specific integrated circuit identifier.

25. The system as in Claim 23, wherein said display adapter includes a graphics processor.

26. The system as in Claim 25, wherein the device identifier includes a graphics processor identifier.

27. The system as in Claim 23, wherein said device-independent driver code includes two-dimensional graphics functions.

28. The system as in Claim 23, wherein the device-specific driver code includes direct 3D functions.

29. The system as in Claim 23, wherein the device-specific driver code includes direct draw functions.

30. The system as in Claim 23, wherein individual executable images of the plurality of executable images include functions unique to a particular device.

31. A computer readable medium tangibly embodying a plurality of programs of instructions, the plurality of programs including:
a set of device-independent functions to support a plurality of different display devices;
a plurality of device-specific driver portions, wherein each device-specific driver portion of the plurality of device-specific driver portions include functions to support only a portion of the plurality of different display devices;
a first function to request a device identifier, wherein the device identifier is capable of identifying a particular display device of the plurality of different display devices; and
a second function to load a particular device-specific driver portion into kernel mode memory, wherein the particular device-specific driver portion is associated with the particular display device of the plurality of different display devices.

32. The computer readable medium as in Claim 31, wherein the second function includes a call to an EngLoadImage function.

33. The computer readable medium as in Claim 32, further including a third function to determine addresses associated with functions of the particular device-specific driver portion.

34. The computer readable medium as in Claim 33, wherein the third function includes a function call to an EngFindImageProcAddress function.

35. The computer readable medium as in Claim 31, wherein the device identifier includes an application specific integrated circuit identifier.

36. The computer readable medium as in Claim 31, further including a table linking device identifiers to individual device-specific driver portions of the plurality of device-specific driver portions.